UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/069,229	12/11/2002	Gilbert Wolrich	10559-305US1	1429

20985 7590 04/28/2009

FISH & RICHARDSON, PC
P.O. BOX 1022
MINNEAPOLIS, MN 55440-1022

EXAMINER	
JOO, JOSHUA	

ART UNIT	PAPER NUMBER
2454	

NOTIFICATION DATE	DELIVERY MODE
04/28/2009	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

PATDOCTC@fr.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte GILBERT WOLRICH, MATTHEW J. ADILETTA,
WILLIAM R. WHEELER, DEBRA BERNSTEIN,
and DONALD F. HOOPER

Appeal 2008-2568
Application 10/069,229
Technology Center 2100

Decided:[1] April 24, 2009

Before HOWARD B. BLANKENSHIP, ST. JOHN COURTENAY III, and THU A. DANG, *Administrative Patent Judges*.

COURTENAY, *Administrative Patent Judge*.

DECISION ON APPEAL

---

[1] The two-month time period for filing an appeal or commencing a civil action, as recited in 37 C.F.R. § 1.304, begins to run from the decided date shown on this page of the decision. The time period does not run from the Mail Date (paper delivery) or Notification Data (electronic delivery).

STATEMENT OF THE CASE

This is a decision on appeal under 35 U.S.C. § 134(a) from the Examiner's rejection of claims 1-21. We have jurisdiction under 35 U.S.C. § 6(b). We AFFIRM.

INVENTION

The invention on appeal is relates generally to branch instructions. (Spec. 1). More particularly, Appellants' invention relates to a branch instruction that branches on a byte being equal or not equal to a byte specified by the branch instruction (*id.* at 13, ll. 1-5).

ILLUSTRATIVE CLAIM

Claim 1, which further illustrates the invention, follows:

> 1. A method of operating a processor comprising:
>
> executing a branch instruction that causes a processor to branch from executing a first sequential series of instructions to a different sequential series of instructions based on a byte, specified by the branch instruction, in a register, specified by the branch instruction, being equal or not equal to a byte value specified by the branch instruction.

PRIOR ART

The Examiner relies upon the following references as evidence in support of the rejections:

Carron	US 4,724,521	Feb. 9, 1988
Bruckert	US 4,742,451	May 3, 1988
Gusefski	US 5,202,972	Apr. 13, 1993
White	US 5,748,950	May 5, 1998

Yates	US 5,802,373	Sep. 1, 1998
Atkins	US 5,898,866	Apr. 27, 1999
Rodriguez	US 6,139,199	Oct. 31, 2000

THE REJECTIONS

Claims 1-3, 6, 10, and 13-21 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over the combination of Carron, Yates, and White.

Claims 4 and 5 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over the combination of Carron, White, Yates, and Atkins.

Claims 7, 8, and 11 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over the combination of Carron, White, Yates, and Bruckert.

Claim 9 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over the combination of Carron, White, Yates, and Gusefski.

Claim 12 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over the combination of Carron, White, Yates, and Rodriguez.

GROUPING OF CLAIMS

(1) Appellants argue claims 1-3, 6, 10, and 13-21 as a group. (App. Br. 10-15). Appellants also argue that claims 9 and 12 are patentable for the same reasons as previously discussed regarding claim 1 (*id.* at 21-22). We will, therefore, treat claims 1-3, 6, 9, 10, and 12-21 as standing or falling with representative claim 1.

(2) Appellants argue claims 4 and 5 as a group. (*id.* at 15-17). We will, therefore, treat claims 4 and 5 as standing or falling with representative claim 4.

(3) Appellants argue claims 7 and 11 as a group (*id.* at 17-19). We will, therefore, treat claims 7 and 11 as standing or falling with claim representative claim 7.

(4) Appellants argue claim 8 separately (*id.* at 19-21).

We accept Appellants' grouping of the claims. *See* 37 C.F.R. § 41.37(c)(1)(vii) ("Notwithstanding any other provision of this paragraph, the failure of appellant to separately argue claims which appellant has grouped together shall constitute a waiver of any argument that the Board must consider the patentability of any grouped claim separately.").

APPELLANTS' CONTENTIONS

Regarding the aforementioned groups 1-4, Appellants contend that certain claim limitations are not disclosed by the Examiner proffered combinations of the aforementioned references. (App. Br. 10-22). We have enumerated the specific claim limitations argued by Appellants in the ISSUES section *infra*. We address Appellants' specific arguments for each issue in our ANALYSIS section *infra.* In addition, Appellants contend that the Examiner has improperly combined the Carron, Yates, and White references with respect to the claims in group 1, *supra* (*id.* at 10-15).

EXAMINER'S RESPONSE

In the "Response to Arguments" section of the Answer, the Examiner maintains that the limitations argued by Appellants are taught and/or suggested by the cited combination of references. (Ans. 11). The Examiner states, *inter alia*, that Appellants have misinterpreted the Examiner's rejection of claim 1, as follows:

> As set forth in the office action dated 5/2/2006, Carron teaches of a COMP_CHAR command, wherein the COMP_CHAR, "Compare[s] byte at specified position in the designated buffer with the byte constant in the command. Branch if the test passes." (Col 134, lines 29-34) Carron teaches of a command comprising instructions for compare and branch. However, Carron does not specifically teach of a single instruction for performing compare and branch. Therefore, the reference White was combined with Carron to specifically teach the feature, wherein White teaches of a compare-and-branch type instruction, which "combines compare and branch operations into one instruction."(Col 5, lines 26-29). Yates was used because Carron does not specifically teach the byte constant in the command is compared to a byte in the register. Yates teaches an instruction 884a that performs a byte compare of register to AL to the constant 3", and instruction 884b that "performs a branch if the value contained in the register AL is not equal to 3 (Col 77, lines 29- 32). Even though Yates teaches of separate compare and branch instructions, White taught that compare and branch instructions can be combined into one instruction.

(*Id.*, alteration in original, underlining omitted).

The Examiner also maintains that the references relied upon have been properly combined (*id.* at 11-13).

ISSUES

We consider the following issues that flow from the contentions of the Appellants and the Examiner:

1. Have Appellants shown the Examiner erred in finding that the combination of Carron, Yates, and White teaches and/or

suggests "executing a branch instruction . . . based on a byte, specified by the branch instruction, in a register, specified by the branch instruction, being equal or not equal to a byte value specified by the branch instruction"? (*See* representative claim 1; *see also* App. Br. 10).

2. Have Appellants shown the Examiner erred by improperly combining the Carron, Yates, and White references regarding the claims in group 1 *supra*? (App. Br. 10-15).
3. Have Appellants shown the Examiner erred in finding that the combination of Carron, White, Yates, and Atkins teaches and/or suggests "an optional token that is set by a programmer and specifies a number i of instructions to execute following the branch instruction before performing the branch operation"? (*See* representative claim 4; *See also* App. Br. 16-17).
4. Have Appellants shown the Examiner erred in finding that the combination of Carron, White, Yates, and Bruckert teaches and/or suggests "an optional token that is set by a programmer and which specifies a guess_branch prefetch for the instruction for the 'branch taken' condition rather than the next sequential instruction"? (*See* representative claim 7; *See also* App. Br. 18-19).

PRINCIPLES OF LAW

"What matters is the objective reach of the claim. If the claim extends to what is obvious, it is invalid under § 103." *KSR Int'l Co. v. Teleflex, Inc.*, 550 U.S. 398, 419 (2007). To be nonobvious, an improvement must be "more than the predictable use of prior art elements according to their established functions." *Id.* at 417.

> Invention or discovery is the requirement which constitutes the foundation of the right to obtain a patent . . . unless more ingenuity and skill were required in making or applying the said improvement than are possessed by an ordinary mechanic acquainted with the business, there is an absence of that degree of skill and ingenuity which constitute the essential elements of every invention.

Dunbar v. Myers, 94 U.S. 187, 197 (1876) (citing *Hotchkiss* v. *Greenwood*, 52 U.S. 248, 267 (1850)) (*Hotchkiss* v. *Greenwood* was cited with approval by the Supreme Court in *KSR*, 550 U.S. at 406, 415, 427).

Appellants have the burden on appeal to the Board to demonstrate error in the Examiner's position. *See In re Kahn*, 441 F.3d 977, 985-86 (Fed. Cir. 2006). Therefore, we look to Appellants' Briefs to show error in the Examiner's proffered prima facie case.

FINDINGS OF FACT

In our analysis *infra*, we rely on the following findings of fact (FF) that are supported by a preponderance of the evidence:

THE PRIMARY CARRON REFERENCE

1. Carron teaches a COMP_BYTE instruction, as follows:

```
                                COMP_BYTE
Compare byte at specified position in designated buffer with the
value stored in the specified variable. Branch if test passes.
Continue with the next command if the test fails. Table 3
contains a list of the comparison operators.

_____________________________________________

    COMP_BYTE <buffer> <position> <operator>
                <variable> <Addr>
    Opcode Group:
                1
    Opcode Number:
                24
    Source Syntax:
                BXXVA
    Object Seq.:
                2-4-1-3-5
    Object Syntax:
                XVBXA
    Entry Point:
                zCMBYT
    Hex Opcode #:
                18
    <buffer>    Destination buffer.
    <position>  Position in the destination buffer to
                be be compared. First char = 1.
    <operator>  Type of comparison requested. (Table 3)
    <variable>  Address of eight bit variable.
    <Addr>      Branch address if test passes.
    _____________________________________________
```

(Carron, col. 134, ll. 6-26).

THE WHITE REFERENCE

2. White teaches that "the present invention microprocessor supports a compare-and-branch (COBR) type instruction. This is an encoding optimization which combines compare and branch operations into one instruction." (Col. 5, ll. 25-29; *see also* the format of the COBR instruction shown in Figure 3).

3. White teaches that

> [w]hen a COBR instruction is executed, the microprocessor compares the source 2 and source 1 operands provided with the instruction and sets a condition code in a register located in the

> instruction sequencer according to the comparison results. Subsequently, a portion of the instruction's opcode is compared against the actual condition code. If the comparison results in a match, then the processor branches to an instruction specified by the displacement value propagated with the COBR instruction. Otherwise, the processor goes to the next sequential instruction.

(Col. 5, ll. 44-52, bolding omitted); *see also* the format of the COBR instruction shown in Figure 3).

THE ATKINS REFERENCE

4. Atkins teaches that "[t]he SETLOOP instruction initializes the loop control hardware REG N [that] serves as the counter to count the number of iterations to execute the loop. It is decremented with each iteration to control the branch at the bottom of the loop." (Col. 9, l. 66 through col. 10, l. 2).
5. Atkins teaches that the "BOT" parameter for the SETLOOP instruction contains the number of instructions within the loop. (Col. 10, ll. 2-3).

THE BRUCKERT REFERENCE

6. Bruckert teaches that

> [t]he invention enables the processor to determine if an instruction is a conditional branch instruction, in which a determination is made as to whether a branch in an instruction stream should or should not be taken depending on the results of prior processing, and to prefetch both the instructions in the "branch taken" instruction stream as well as from the "branch not taken" instruction stream.

(Col. 1, ll. 22-29).

7. Bruckert teaches that

> [t]he execution unit transmits signals to the fetch unit indicating whether or not the branch is to be taken. If the branch is not taken, the fetch unit causes the memory to abort the "branch taken" instruction prefetch. If the execution unit informs the fetch unit that the branch is to be taken, the fetch unit allows the memory to complete the prefetch. The operand fetch portion of the fetch unit then transfers the address of the location of the "branch taken" instruction stream to the instruction prefetch portion, and the fetch unit then performs as before the conditional branch instruction, using the "branch taken" instruction stream.

(Col. 2, ll. 57-68).

ANALYSIS

At the outset, we consider Appellants' arguments in the Briefs only to the extent that such arguments are directed to claimed subject matter.

ISSUE 1

We decide the question of whether Appellants have shown the Examiner erred in finding that the combination of Carron, Yates, and White teaches and/or suggests "executing a branch instruction . . . based on a byte, specified by the branch instruction, in a register, specified by the branch instruction, being equal or not equal to a byte value specified by the branch instruction." (*See* representative claim 1; *see also* App. Br. 10).

Appellants contend that the combination of Carron, Yates, and White does not teach the aforementioned limitations regarding representative claim 1 (App. Br. 10).

Based upon our review of the record before us, we find both the Examiner and the Appellants have not fully appreciated the teaching value of the Carron and White references, particularly when each reference is considered alone.

Based upon our review of the Carron reference, we disagree with the Examiner's statement that "Carron does not specifically teach of a single instruction for performing compare and branch." (Ans. 11, underlining omitted). A close examination of column 134 of Carron reveals that the COMP_BYTE instruction performs not just a comparison, but also a branch operation (FF 1). As taught by Carron, the "byte at [the] specified position in [a] designated buffer" is compared with "the value stored in a specified variable" and a branch is taken if the comparison test passes (FF 1).

Therefore, we find Carron anticipates the argued limitations of claim 1 of "executing a branch instruction . . . based on a byte, specified by the branch instruction, in a register, specified by the branch instruction, being equal or not equal to a byte value specified by the branch instruction."

We also find that the White reference anticipates the argued limitations of representative claim 1. White expressly teaches a compare-and-branch (COBR) type instruction that is described as an encoding optimization which *combines compare and branch operations into one instruction.* (FF 2). White teaches that

> [w]hen a COBR instruction is executed, the microprocessor compares the source 2 and source 1 operands provided with the instruction and sets a condition code in a register located in the instruction sequencer according to the comparison results. Subsequently, a portion of the instruction's opcode is compared against the actual condition code. *If the comparison results in a match, then the processor branches to an instruction specified by the displacement value propagated with the*

> *COBR instruction*. Otherwise, the processor goes to the next sequential instruction.

(FF 3, emphasis added). Therefore, we find White anticipates the argued limitations of "executing a branch instruction . . . based on a byte, specified by the branch instruction, in a register, specified by the branch instruction, being equal or not equal to a byte value specified by the branch instruction." (Claim 1).

A disclosure that anticipates under 35 U.S.C. § 102 also renders the claim unpatentable under 35 US.C. § 103, for "anticipation is the epitome of obviousness." *Jones v. Hardy,* 727 F.2d 1524, 1529 (Fed. Cir. 1984). *See also In re Fracalossi,* 681 F.2d 792, 794 (CCPA 1982); *In re Pearson,* 494 F.2d 1399, 1402 (CCPA 1974). In affirming a multiple reference rejection under 35 U.S.C. § 103, the Board may rely on one reference alone in an obviousness rationale without designating it as a new ground of rejection. *In re Bush*, 296 F.2d 491, 496 (CCPA 1961); *In re Boyer*, 363 F.2d 455, 458 n.2 (CCPA 1966).

ISSUE 2

We decide the question of whether Appellants have shown the Examiner erred by improperly combining the Carron, Yates, and White references with respect to the claims in group 1 *supra* (App. Br. 10-15).

As discussed above, we have found that either one of the Carron or White references, when considered individually, anticipates the argued limitations of representative claim 1. Because "anticipation is the epitome of obviousness," we find Appellants' arguments directed to the Examiner's

proffered combination of Carron, Yates, and White to be misdirected. *See Jones,* 727 F.2d at 1529.

Nevertheless, we find Appellants have provided no evidence to show that combining the respective familiar elements of Carron, Yates, and White (i.e., compare and branch instructions) in the manner proffered by the Examiner was "uniquely challenging or difficult for one of ordinary skill in the art" (*see Leapfrog Enters., Inc. v. Fisher-Price, Inc.*, 485 F.3d 1157, 1162 (Fed. Cir. 2007) (citing *KSR*, 550 U.S. at 418)).

In *KSR*, the Supreme Court emphasized "the need for caution in granting a patent based on the combination of elements found in the prior art" and discussed circumstances in which a patent might be determined to be obvious. *Id.* at 401 (citing *Graham v. John Deere Co.*, 383 U.S. 1, 12 (1966)) (citation omitted). The Court reaffirmed principles based on its precedent that "[t]he combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results." *Id.* The operative question in this "functional approach" is thus "whether the improvement is more than the predictable use of prior art elements according to their established functions." *Id.* at 414, 401.

Here, we find the Examiner's proffered combination of Carron, Yates, and White is merely a combination of familiar elements (i.e., compare and branch instructions) according to known methods that yields a predictable result. Therefore, we find Appellants' "teaching away," "changed principle of operation," and "unsatisfactory for its intended purpose" arguments unavailing. We note that in the Briefs Appellants do not separately contest the combinability of the references for the remaining claims in groups 2-4, as defined *supra*.

ISSUE 3

We decide the question of whether Appellants have shown the Examiner erred in finding that the combination of Carron, White, Yates, and Atkins teaches and/or suggests "an optional token that is set by a programmer and specifies a number i of instructions to execute following the branch instruction before performing the branch operation." (*See* representative claim 4; *See also* App. Br. 16-17).

Appellants contend that the parameter specified in Atkins' "SETLOOP instruction has no relevance to the claim feature [claim 4], but rather indicates the number of instructions before a branch instruction. In contrast, Appellant's claimed feature specifies the number of instructions to execute after the branch instruction." (App. Br. 17).

We disagree. We note that Atkins teaches that "[t]he SETLOOP instruction initializes the loop control hardware REG N [that] serves as the counter to count the number of iterations to execute the loop. It is decremented with each iteration to control the branch at the bottom of the loop." (FF 4). Atkins further teaches that the "BOT" parameter for the SETLOOP instruction contains the number of instructions within the loop (FF 5).

Therefore, for essentially the same reasons argued by the Examiner, we find that Atkins teaches and/or suggests "an optional token [SETLOOP INSTRUCTION] that is set by a programmer and specifies a number i of instructions [specified by the "BOT" parameter] to execute following the branch instruction before performing the branch operation." (*See* representative claim 4). We note that the branch condition is tested, as taught by Atkins, at the bottom of the loop (FF 4). We find that a particular

number of instructions (as specified by the "BOT" parameter) are executed before and after the branch test condition until the loop condition terminates.

ISSUE 4

We decide the question of whether Appellants have shown the Examiner erred in finding that the combination of Carron, White, Yates, and Bruckert teaches and/or suggests "an optional token that is set by a programmer and which specifies a guess_branch prefetch for the instruction for the 'branch taken' condition rather than the next sequential instruction." (*See* representative claim 7; *See also* App. Br. 18-19).

Regarding the Bruckert reference (relied on by the Examiner), Appellants contend that "[b]ecause *both* [the] 'branch taken' and 'branch not taken' instruction sequences are retrieved, Bruckert would have no need to include a token that would indicate which instruction should be retrieved following the execution of a branch instruction." (App. Br. 18, emphasis added).

In response, the Examiner acknowledges that Brucket does not explicitly teach a 'token' for specifying a prefetch for the 'branch taken' instruction (Ans. 16). However, the Examiner finds that a command or an identifier such as a token would be needed to indicate the prefetch for the 'branch taken' instruction (*Id.*). The Examiner further finds that the prefetch for 'branch taken' or the indication for the prefetch (e.g. command, identifier, or token or the like) is optional in that a programmer has the option to include the specific prefetch as part of the programming. The Examiner also finds that Appellants' feature of an optional token in the claim does not define what is optional about the token. (*Id.*).

We note that Bruckert teaches that

> [t]he invention enables the processor to determine if an instruction is a conditional branch instruction, in which a determination is made as to whether a branch in an instruction stream should or should not be taken depending on the results of prior processing, and to prefetch both the instructions in the "branch taken" instruction stream as well as from the "branch not taken" instruction stream.

(FF 6). Bruckert further teaches that

> [t]he execution unit transmits signals to the fetch unit indicating whether or not the branch is to be taken. If the branch is not taken, the fetch unit causes the memory *to abort the "branch taken" instruction prefetch. If the execution unit informs the fetch unit that the branch is to be taken, the fetch unit allows the memory to complete the prefetch.*

(FF 7, emphasis added).

Because Bruckert teaches that the *"branch taken" instruction prefetch is aborted if* the branch is not taken, we find the weight of the evidence supports the Examiner's position. Therefore, we find that the Examiner's proffered combination of Carron, White, Yates, and Bruckert teaches and/or suggests "an optional token that is set by a programmer and which specifies a guess_branch prefetch for the instruction for the 'branch taken' condition rather than the next sequential instruction." (*See* representative claim 7; *See also* App. Br. 18-19).

CONCLUSION

In summary, we find Appellants' arguments directed to each of representative claims 1, 4, and 7 to be unavailing for the reasons set forth above regarding ISSUES 1-4. While Appellants have argued claim 8 separately, we note that claim 8 merely combines the limitations of representative claim 4 and representative claim 7 (ISSUES 3, 4). Therefore, we also find unavailing Appellants arguments directed to claim 8.

Claims 2-3, 6, 10, and 13-21 fall with representative claim 1 as being unpatentable over the combination of Carron, Yates, and White.

Claim 9 falls with representative claim 1 as being unpatentable over the combination of Carron, White, Yates, and Gusefski.

Claim 12 falls with representative claim 1 as being unpatentable over the combination of Carron, White, Yates, and Rodriguez.

Claim 5 falls with representative claim 4 as being unpatentable over the combination of Carron, White, Yates, and Atkins.

Claims 8 and 11 fall with representative claim 4 and 7, as discussed *supra*.

Given our findings of facts and analysis set forth above, we find Appellants have not sustained the requisite burden on appeal of providing arguments or evidence persuasive of error in the Examiner's rejections of claims 1-21 as being unpatentable under 35 U.S.C. § 103(a).

DECISION

We affirm the Examiner's decision rejecting claims 1-21 under 35 U.S.C. § 103(a).

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a)(1)(iv).

AFFIRMED

msc

FISH & RICHARDSON, PC
P.O. BOX 1022
MINNEAPOLIS MN 55440-1022